Center for Advanced Computation

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

CAC Document No. 110

AN ALGORITHM FOR THE SYMMETRIC GENERALIZED EIGENVALUE PROBLEM $Ax = \lambda Bx$

by

Chang-Chung Chang

February 14, 1974

CAC Document No. 110

AN ALGORITHM FOR THE SYMMETRIC GENERALIZED EIGENVALUE PROBLEM $Ax = \lambda Bx$

by

Chang-Chung Chang

Center for Advanced Computation
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

February 14, 1974

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense as monitored by the U. S. Army Research Office-Durham under Contract DAHC04-72-C-0001, and was submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science in the Graduate College of the University of Illinois at Urbana-Champaign, 1974.

ABSTRACT

An SQZ algorithm is developed, in a way similar to that of Moler and Stewart's QZ method, for handling symmetric A and B where B is an ill-conditioned positive definite matrix. This algorithm preserves symmetry, reduces the storage requirements, uses less time, and produces only real eigenvalues.

ACKNOWLEDGEMENT

The author would like to express his deepest thanks to his advisor. Professor Ahmed H. Sameh, for his guidance and encouragement during this research. Without his encouragement and patience, this work would have never been done. The author would also like to thank Mrs. Lois Pelczar for her excellent typing of this manuscript.

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense as monitored by the U.S. Army Research Office under Contract No. DAHC04-72-C-0001, in part by the Department of Computer Science, and in part by the Center for Advanced Computation of the University of Illinois at Urbana-Champaign.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In many applications, such as those in physical sciences, the solution of the generalized eigenvalue problem $Ax = \lambda Bx$ is often required, where A and B are real symmetric matrices and B is positive definite. There exist several methods for solving this problem. The well-known Cholesky-Wilkinson method [1] uses Cholesky factorization of B, $B = LL^t$, to reduce the problem into standard form. However, this method requires inverting the factors of B which may lead to a bad solution if B is ill-conditioned. For a nearly singular B, Peters and Wilkinson [4] describe an algorithm which approximates the null space of B and removes it from the problem to get a well-conditioned problem. This method involves determining the rank of B. If a wrong decision is made, the well-conditioned eigenvalues may be seriously affected. In [6], Fix and Heiberger designed an algorithm which is a variant of the Peters-Wilkinson method [4] for nearly semidefinite B, i.e. is concerned with the case when B (or A and B) is ill-conditioned with respect to inversion and B is permitted to be positive semidefinite. Peters and Wilkinson also describe another efficient algorithm in [2] for the calculation of specified eigenvalues of $Ax = \lambda Bx$ with band symmetric A and B, the latter being positive definite. In their method, every eigenvalue is isolated using the Sturm sequence property of leading principal minors of $A - \lambda B$ and is then computed accurately using a modified version of successive linear interpolation. Recently, Moler

and Stewart have presented the QZ method [9] which was designed primarily for nonsymmetric matrices A and B, and does not require inversion of B. If A and B were symmetric, this algorithm destroys symmetry and requires more arithmetic operations and storage and may even produce complex eigenvalues.

In this paper, our SQZ algorithm is developed in a similar way as the QZ for handling symmetric A and B where B is an ill-conditioned positive definite matrix. This algorithm preserves symmetry, reduces the storage requirements, uses less time, and produces only real eigenvalues. Since our method is actually a symmetric case of Moler and Stewart's QZ method, we will call our algorithm "SQZ". The algorithm is based on the following observations:

1. For matrices A and B if $A = ZBZ^t$ for some non-singular matrix Z, then the matrices A and B are called congruent. If $A_2 = ZA_1Z^t$ and $B_2 = ZB_1Z^t$ then the generalized eigenvalue problems $A_1x = \lambda B_1x$ and $A_2y = \lambda B_2y$ have the same eigenvalues, and the eigenvectors are related by $x = Z^ty$.
2. For matrices A and B, there exist unitary matrices U and V such that both $A' = U^HAV$ and $B' = U^HBV$ are upper triangular. The values a'_{ii}/b'_{ii} are the eigenvalues λ_i in $Ax = \lambda Bx$. [3]
3. If A and B are symmetric with B positive definite, then there exists a matrix U satisfying $U^HBU = I$ such that $A' = U^HAU$ is diagonal. [3]

The algorithm consists of the following four stages:

(i) B is reduced to a diagonal matrix (an iterative process), while the updated A is still symmetric. This stage requires only

orthogonal transformations.

(ii) A is reduced to the tridiagonal form keeping B diagonal. This stage requires both orthogonal and elementary transformations.

(iii) A is subjected to the QR transformations with elementary transformations to keep B diagonal, an iterative process.

(iv) After several iterations in (iii), A approaches the diagonal form and the eigenvalues λ_i will be given by a_{ii}/b_{ii} if $b_{ii} \neq 0$. If $a_{ii} \neq 0$ and $b_{ii} = 0$, then we will have an infinite eigenvalue λ_i. If both $a_{ii} = b_{ii} = 0$, then any scalar can be an eigenvalue of $Ax = \lambda Bx$.

Stabilized elementary transformations are used throughout the algorithm.

CHAPTER 2

SQZ ALGORITHM

2.1. Theoretical Basis

Moler and Stewart's QZ algorithm is a generalization of Francis' QR method [11] to solve the problem $Ax = \lambda Bx$ where A and B are general square matrices. If both A and B are real symmetric matrices, the QZ algorithm will destroy symmetry and hence requires more time and storage and may produce nonreal eigenvalues. This is not economical or practical, especially for matrices of large size. As we mentioned previously, several algorithms have been developed for dealing with the above problem. However, it was mainly assumed that B in addition of being positive definite is also well-conditioned. In this paper we present an algorithm to deal economically with the problem when B is ill-conditioned. The algorithm requires much less storage and time, and produces only real eigenvalues.

Before we go into the details of our algorithm, which we will call SQZ, let us present the idea of the QR method for the eigenvalue problem $Cx = \lambda x$,

1. Reduce C to the upper Hessenberg form using similarity transformations.

2. Find an origin shift λ using the roots of the lower right-hand 2×2 principal submatrix of C.

3. Find an orthogonal matrix Q such that $Q(C - \lambda I) = R$, where R is upper triangular.

4. Let $C = QCQ^t$, then the matrix C is upper Hessenberg again.

5. If the off-diagonal elements of C are not negligible, then go back to 2.

6. The eigenvalues of the original matrix are the diagonal elements of C.

If the original matrix C is real symmetric, it is first reduced to the tridiagonal form which is preserved by the QR transformations.

Moler and Stewart's QZ algorithm is motivated by the QR method described above. We present the idea of this algorithm:

1. Reduce simultaneously A to upper Hessenberg form and B to triangular form.

2. Find the origin shift using the roots of $\tilde{A}y = \lambda\tilde{B}y$, where $\tilde{A}$ and $\tilde{B}$ are the lower 2×2 principal submatrices of A and B, respectively.

3. Find the orthogonal matrices Q and Z, such that QAZ is upper Hessenberg and $Q(A - \lambda B)$ and QBZ are both upper triangular matrices.

4. Let QAZ be denoted by A, QBZ be denoted by B.

5. If the off-diagonal elements of A are not negligible, then go back to 2.

6. The i^{th} eigenvalue is $\frac{a_{ii}}{b_{ii}}$ if $b_{ii} \neq 0$.

The idea of our SQZ is similarly presented as:

1. Reduce simultaneously A to tridiagonal matrix and B to diagonal.

2. Find the origin shift.

3. Find a transformation L such that LAL^t is tridiagonal, $L(A - \lambda B)$ is upper triangular, and LBL^t is diagonal.

4. Let LAL^t and LBL^t be denoted by A and B, respectively.

5. If the off-diagonal elements of A are not negligible, then go back to 2.

6. The i^{th} eigenvalue is $\frac{a_{ii}}{b_{ii}}$ if $b_{ii} \neq 0$.

To simplify the explanation in our SQZ algorithm, we introduce the following notations:

1. Consider the real symmetric matrix,

$$\tilde{A} = \begin{array}{cc} \left[\begin{array}{c|cccccc} T_{j-1} & & & 0 & & & \\ \hline & x & \cdots & x_1 & x_2 & 0 & \cdots\, 0 \\ & \vdots & & & & & \\ 0 & x_1 & & & & & \\ & x_2 & & & X & & \\ & 0 & & & & & \\ & \vdots & & & & & \\ & 0 & & & & & \end{array}\right] & \begin{array}{l} \\ \leftarrow j^{th} \text{ row} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \end{array} \\ \begin{array}{cccc} \quad\quad \uparrow & & & \uparrow \\ \quad\quad j^{th} \text{ col.} & & & i^{th} \text{ col.} \end{array} & \end{array}$$

by G, we mean the class of orthogonal transformations of the form,

$$G_{i,j} = \begin{array}{cc} \left[\begin{array}{cccccc} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & c & s & & \\ & & & s & -c & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{array}\right] & \begin{array}{l} \\ \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \end{array} \end{array}$$

such that $G_{i,j}\tilde{A}G_{i,j}^{t}$ annihilates elements in the positions (i, j) and (j, i), i.e., eliminates x_2 in the matrix $\tilde{A}$, where c and s are chosen to satisfy the following:

$$\begin{bmatrix} c & s \\ s & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

hence we have,

$$c = \frac{x_1}{r}, \quad s = \frac{x_2}{r}, \quad r = \mathrm{sign}(x_1)\cdot\sqrt{x_1^2 + x_2^2}$$

2. Consider the matrix,

$$D' = \begin{array}{cl} \begin{bmatrix} x & & & & & \\ & \ddots & & & & \\ & & x & & & \\ & & & g_1 & f & & \\ & & & f & g_2 & & \\ & & & & & x & \\ & & & & & & \ddots \\ & & & & & & & x \end{bmatrix} & \begin{array}{l} \\ \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \end{array} \end{array}$$

by E, we mean the class of elementary transformations of the form,

$$E_i = \begin{array}{cl} \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & p & \\ & & 0 & 1 & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} & \begin{array}{l} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \end{array} \end{array}$$

such that $E_i D' E_i^t$ produces 0 in the positions (i-1, i) and (i, i-1), i.e., annihilates the element f in D′ above as follows:

$$\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & p & \\ & & 0 & 1 & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & g_1 & f & \\ & & f & g_2 & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} \begin{array}{cl} \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & 0 & \\ & & p & 1 & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} & \begin{array}{l} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \end{array} \end{array}$$

$$= \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & \tilde{g}_1 & 0 & & \\ & & 0 & g_2 & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \begin{array}{l} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \end{array}$$

where $p = \frac{-f}{g_2}$, and $\tilde{g}_1 = g_1 + pf$.

3. By S, we mean the class of diagonal matrices of the form,

$$S_i = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & s_1 & 0 & & \\ & & & 0 & s_2 & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{bmatrix} \begin{array}{l} \\ \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \end{array}$$

2.2. SQZ ALGORITHM

We will start by describing the algorithm for the generalized eigenvalue problem

$$\tilde{A}\tilde{x} = \lambda\tilde{D}\tilde{x} \qquad (1)$$

where $\tilde{A}$ is a real symmetric matrix and $\tilde{D} = \text{diag}(d_{11}, d_{22}, \ldots, d_{nn})$ in which $0 \leq d_{11} \leq d_{22} \leq \ldots \leq d_{nn}$. Our SQZ algorithm consists of two steps,

STEP I--Reduce $\tilde{A}$ to the tridiagonal form, keeping $\tilde{D}$ diagonal:

Let,

$$\tilde{A} = \begin{bmatrix} x & x \cdots x & x_1 & x_2 \\ x & x \cdots x & x & x \\ \multicolumn{4}{c}{\cdots\cdots\cdots\cdots\cdots\cdots} \\ \multicolumn{4}{c}{\cdots\cdots\cdots\cdots\cdots\cdots} \\ x_1 & x \cdots x & x & x \\ x_2 & x \cdots x & x & x \end{bmatrix} \text{ and } \tilde{D} = \begin{bmatrix} x & & & & & \\ & x & & & & \\ & & \ddots & & & \\ & & & x & & \\ & & & & d_1 & \\ & & & & & d_2 \end{bmatrix}$$

Consider the orthogonal transformation,

$$G_{n,1} = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & & \\ & & & c & s \\ & & & s & -c \end{bmatrix}$$

From (1), we have

$$(G_{n,1}\tilde{A}G^t_{n,1})\,(G_{n,1}\tilde{x}) = (G_{n,1}\tilde{D}G^t_{n,1})\,(G_{n,1}\tilde{x})$$

or, $\quad A'x' = \lambda D'x' \qquad (2)$

The new matrices A' and D' are of the form,

$$A' = \begin{bmatrix} x & x & \cdots & x & x & 0 \\ x & x & \cdots & x & x & x \\ \cdots & & & & & \cdots \\ \cdots & & & & & \cdots \\ x & x & \cdots & x & x & x \\ 0 & x & \cdots & x & x & x \end{bmatrix}, \quad D' = \begin{bmatrix} x & & & & \\ & \ddots & & & \\ & & x & & \\ & & & g_1 & f \\ & & & f & g_2 \end{bmatrix}$$

Now, we choose the elementary transformation,

$$E_n = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & p \\ & & 0 & 1 \end{bmatrix}$$

where $p = \frac{-f}{g_2}$, such that $E_n A' E_n^t$ has the zeros in positions (1, n) and (n, 1) preserved, and $E_n D' E_n^t$ is a diagonal matrix. Thus,

$$(E_n G_{n,1}\tilde{A}G^t_{n,1}E^t_n)\,(E^{-t}_n G_{n,1}\tilde{x}) = \lambda(E_n G_{n,1}\tilde{D}G^t_{n,1}E^t_n)\,(E^{-t}_n G_{n,1}\tilde{x})$$

or, $\quad \tilde{A}'\tilde{x}' = \lambda\tilde{D}'\tilde{x}' \qquad (3)$

where we have,

$$\tilde{D}' = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \tilde{g}_1 & 0 \\ & & & 0 & g_2 \end{bmatrix}$$

in which

$$\tilde{g}_1 = g_1 + pf = \frac{d_{n-1,n-1} d_{nn}}{g_2} > 0 \tag{4}$$

In general, suppose we have $\tilde{A}$ and $\tilde{D}$ as follows:

$$\tilde{A} = \begin{array}{c} \left[\begin{array}{c|cccccc} T_{i-1} & & & 0 & & & \\ \hline & x & \cdots & x_1 & x_2 & 0 \cdots & 0 \\ & \vdots & & & & & \\ & x_1 & & & & & \\ & & & X & & & \\ & x_2 & & & & & \\ & 0 & & & & & \\ & \vdots & & & & & \\ & 0 & & & & & \end{array}\right] \begin{array}{l} \\ \leftarrow j^{th} \text{ row} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \\ \end{array} \\ \quad j^{th} \uparrow \text{col.} \qquad i^{th} \uparrow \text{col.} \end{array} \qquad \tilde{D} = \begin{bmatrix} x & & & & \\ & \ddots & & & \\ & & x & & \\ & & & d_1 & & \\ & & & & d_2 & \\ & & & & & \ddots \\ & & & & & & x \end{bmatrix} \begin{array}{l} \\ \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \end{array}$$

where T_{j-1} is a tridiagonal matrix of order $(j-1)$ and we are going to annihilate x_2 in $\tilde{A}$, so we apply the orthogonal transformation $G_{i,j}$ to (1) and obtain,

$$(G_{i,j}\tilde{A}G^t_{i,j})\ (G_{i,j}\tilde{x}) = (G_{i,j}\tilde{D}G^t_{i,j})\ (G_{i,j}\tilde{x}) \tag{5}$$

where $A' = G_{i,j}\tilde{A}G^t_{i,j}$ and $D' = G_{i,j}\tilde{D}G^t_{i,j}$ are of the form,

$$\begin{bmatrix} T_{j-1} & \vdots & 0 & & & & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ & \vdots & x \; --- \; x_1' & 0 & 0 & --- & 0 \\ 0 & \vdots & x_1' & & & & \\ & \vdots & 0 & & X & & \\ & \vdots & 0 & & & & \\ & \vdots & 0 & & & & \end{bmatrix}, \qquad \text{and} \qquad \begin{bmatrix} x & & & \\ & \ddots & & \\ & & g_1 & f & \\ & & f & g_2 & \\ & & & & \ddots \\ & & & & & x \end{bmatrix}$$

respectively, a non-zero element f is introduced in positions (i, i-1) and (i-1, i) of D'.

Next, we apply an elementary transformation E_i to (5) and obtain,

$$(E_i A' E_i^t)\,(E_i^{-t} \tilde{x}') = (E_i D' E_i^t)\,(E_i^{-t} \tilde{x}') \tag{6}$$

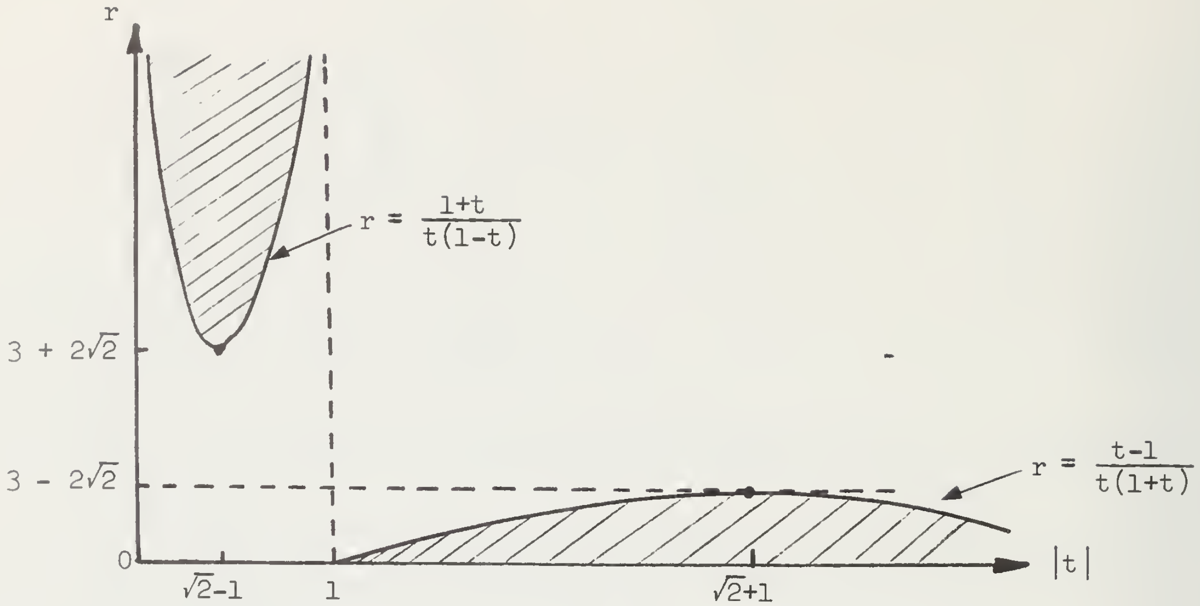
where E_i is such that $E_i A' E_i^t$ has the same zeros as in A' and $E_i D' E_i^t$ is diagonal. The elementary transformations E_i may cause numerical instability if $|p|>1$ (see Wilkinson [10], p. 164). For $|p| = \left|\frac{f}{g_2}\right|$ to be less than or equal to 1, we have,

$$-1 \le \frac{cs(d_1-d_2)}{s^2 d_1 + c^2 d_2} \le 1$$

Let $r = \frac{d_1}{d_2} > 0$, and $t = \frac{x_2}{x_1} = \frac{s}{c}$, then

$$-1 \le \frac{t(r-1)}{rt^2+1} \le 1$$

Hence the region in the $(r, |t|)$-plane in which $|p| > 1$ is the shaded area in the following figure.



Suppose that x_1, x_2 and d_1, d_2 are the elements which yield $|p| > 1$ in $\tilde{A}$ and $\tilde{D}$, respectively. In this case we apply a diagonal transformation S_i in which $s_1 = 1$ and $s_2 = \alpha$. Therefore $S_i\tilde{A}S_i^t$ and $S_i\tilde{D}S_i^t$ yield,

$$\begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ \alpha x_2 \end{bmatrix},$$

and $$\begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \alpha \end{bmatrix} = \begin{bmatrix} d_1 & 0 \\ 0 & \alpha^2 d_2 \end{bmatrix}$$

thus, $\tilde{t} = \alpha t$, and $\tilde{r} = \frac{r}{\alpha^2}$.

One way of choosing an appropriate scale factor α is, for $16^{-2} \leq |t| \leq 16^2$, we use $\alpha = \frac{1}{t}$, i.e. $\tilde{t} = 1$, hence $|p| \leq 1$. For the other values of $|t|$ we use the transformation,

$$\begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} = \begin{bmatrix} \sqrt{d_2} & 0 \\ 0 & \sqrt{d_1} \end{bmatrix}$$

hence $f \equiv 0$ and no elementary transformation is required. Consider the matrices $\tilde{A}$ and $\tilde{D}$, where $16^{-2} \leq |t| \leq 16^{2}$,

$$\tilde{A} = \begin{array}{c} \left[\begin{array}{c|ccccccc} T_{j-1} & & & & 0 & & & \\ \hline & x & -- & x_1 & x_2 & 0 & -- & 0 \\ & \vdots & & & & & & \\ & x_1 & & & & & & \\ & x_2 & & & X & & & \\ & 0 & & & & & & \\ & \vdots & & & & & & \\ & 0 & & & & & & \end{array}\right] \\ \quad j^{th}\uparrow\text{col.} \quad i^{th}\uparrow\text{col.} \end{array} \begin{array}{l} \\ \leftarrow j^{th} \text{ row} \\ \\ , \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \\ \end{array} \qquad \tilde{D} = \begin{bmatrix} x & & & \\ & \ddots & & \\ & & d_1 & \\ & & & d_2 & \\ & & & & \ddots \\ & & & & & x \end{bmatrix} \begin{array}{l} \\ \\ \\ \leftarrow i^{th} \text{ row} \\ \\ \\ \end{array}$$

and the corresponding matrix, (after performing orthogonal transformation),

$$D' = \begin{bmatrix} x & & & & \\ & \ddots & & & \\ & & g_1 & f & \\ & & f & g_2 & \\ & & & & \ddots & \\ & & & & & x \end{bmatrix} \begin{array}{l} \\ \\ \\ - i^{th} \text{ row} \\ \\ \\ \end{array}$$

in which $\left|\frac{f}{g_2}\right| > 1$. Consider the diagonal transformation,

$$S_i = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & & \\ & & & \alpha & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \begin{array}{l} \\ \\ \\ - i^{th} \text{ row} \\ \\ \\ \end{array}$$

where $\alpha = \frac{1}{t} = \frac{x_1}{x_2}$, thus

$$S_i \tilde{A} S_i^t = \left[\begin{array}{c|ccccccc} T_{j-1} & & & & 0 & & & \\ \hline & x & --- & x_1 & \alpha x_2 & 0 & --- & 0 \\ 0 & \vdots & & & & & & \\ & x_1 & & & & & & \\ & \alpha x_2 & & & \tilde{X} & & & \\ & 0 & & & & & & \\ & 0 & & & & & & \end{array}\right]$$

and, $$S_i\tilde{D}S_i^t = \begin{bmatrix} x & & & & & \\ & \ddots & & & & \\ & & d_1 & & & \\ & & & \alpha^2 d_2 & & \\ & & & & \ddots & \\ & & & & & x \end{bmatrix}$$

Therefore the multiplier p in E_i will be less than or equal to 1 in absolute value,

$$|p| = \left|\frac{f}{g_2}\right| = \left|\frac{cs(d_1-\alpha^2 d_2)}{s^2 d_1 + c^2 d_2}\right| = \left|\frac{t^2 r-1}{t^2 r+1}\right| \leq 1,$$

since $r > 0$, and $\alpha = \frac{1}{t}$. For those cases when $\left|\frac{f}{g_2}\right| \leq 1$, we choose $S_i \equiv I$.

Now if we define,

$$Z_j = V_{j+2}\, V_{j+1} \cdots V_{n-1}\, V_n, \qquad j = 1, 2, \ldots, n-2 \quad (7)$$

where $V_i = E_i G_{i,j} S_i$, then we have,

$$Z_{n-2}Z_{n-1}\cdots Z_2 Z_1 \tilde{A} Z_1^t Z_2^t \cdots Z_{n-1}^t Z_n^t = T \qquad \text{(tridiagonal)},$$

and $$Z_{n-2}Z_{n-1}\cdots Z_2 Z_1 \tilde{D} Z_1^t Z_2^t \cdots Z_{n-1}^t Z_n^t = D \qquad \text{(diagonal)}.$$

That is, by the transformation $Z = Z_{n-2}\cdots Z_2 Z_1$, we will be dealing with the problem $Ty = \lambda Dy$ instead of the original problem $\tilde{A}\tilde{x} = \lambda\tilde{D}\tilde{x}$, where

$$T = Z\tilde{A}Z^t$$

$$D = Z\tilde{D}Z^t$$

$$\tilde{x} = Z^t y.$$

To reduce $\tilde{A}$ to T and $\tilde{D}$ to D we require roughly $\frac{4}{3}n^3$ multiplications and $\frac{1}{2}n^2$ square roots.

<u>STEP II</u>--<u>Reduce T to Diagonal and preserve the diagonal D</u>:

This is an iterative process. The origin shift is obtained from the lowest 2×2 principal submatrix of $D^{-1/2}TD^{-1/2}$. Let us assume that the lowest 2×2 principal submatrix of T and the corresponding 2×2 principal

submatrix of D are given by

$$\begin{bmatrix} \alpha_1 & \beta_2 \\ \beta_2 & \alpha_2 \end{bmatrix} \text{ and } \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix},$$

respectively. The shift λ is chosen as the eigenvalue of

$$\begin{bmatrix} \alpha_1 & \beta_2 \\ \beta_2 & \alpha_2 \end{bmatrix} y = \lambda \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} y \tag{8}$$

or

$$\begin{bmatrix} \tau_1 & \gamma \\ \gamma & \tau_2 \end{bmatrix} \tilde{y} = \lambda \tilde{y} \tag{9}$$

where

$$\begin{aligned} \tau_1 &= d_1^{-1}\alpha_1 \\ \tau_2 &= d_2^{-1}\alpha_2 \\ \gamma &= d_1^{-1/2} d_2^{-1/2} \beta_2 \\ \tilde{y} &= D^{1/2} y \end{aligned} \tag{10}$$

which is closer to τ_2. Thus λ is given by,

$$\lambda = \tau_2 + \frac{\gamma}{q \pm v}$$

where

$$q = \frac{\tau_2 - \tau_1}{\gamma} \tag{11}$$

$$v = \sqrt{q^2 + 1}$$

and the sign in $q \pm v$ is chosen to yield,

$$|q \pm v| = |q| + v.$$

Now, let the matrices in $Ty = \lambda Dy$ be given by,

$$T = \begin{bmatrix} x_1 & x_2 & & & \\ x_2 & x & x & & \\ & x & x & x & \\ & & \ddots\, x & \ddots\, x & \ddots\, x \\ & & & x & x \end{bmatrix}, \quad D = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & x & \\ & & & \ddots \\ & & & & x \end{bmatrix}$$

we determine an orthogonal transformation $G_{2,1}$ to produce

$$\begin{bmatrix} c & s \\ s & -c \end{bmatrix} \begin{bmatrix} x_1 - \lambda d_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 \\ 0 \end{bmatrix} \tag{12}$$

and apply it to T and D,

$$(G_{2,1} T G^t_{2,1})\,(G_{2,1} y) = \lambda (G_{2,1} D G^t_{2,1})\,(G_{2,1} y)$$

Thus $T' = G_{2,1} T G^t_{2,1}$ and $D' = G_{2,1} D G^t_{2,1}$ are of the form,

$$T' = \begin{bmatrix} x & x & + & & \\ x & x & x & & \\ + & x & x & x & \\ & & \ddots\, x & \ddots\, x & \ddots\, x \\ & & & x & x \end{bmatrix}, \quad D' = \begin{bmatrix} g_1 & f & & \\ f & g_2 & & \\ & & x & \\ & & & \ddots \\ & & & & x \end{bmatrix}.$$

Next, find an elementary transformation E_2 to annihilate f but does not introduce any new nonzero elements in T', thus $\tilde{T} = E_2 T' E^t_2$ and $\tilde{D} = E_2 D' E^t_2$ are of the form,

$$\tilde{T} = \begin{bmatrix} x & x_1 & x_2 & & \\ x_1 & x & x & & \\ x_2 & x & x & x & \\ & & \ddots\, x & \ddots\, x & \ddots\, x \\ & & & x & x \end{bmatrix}, \text{ and } \tilde{D} = \begin{bmatrix} x & & & & \\ & d_1 & & & \\ & & d_2 & & \\ & & & x & \\ & & & & \ddots\, x \end{bmatrix}.$$

Then by another orthogonal transformation $G_{3,1}$, we annihilate x_2 in $\tilde{T}$ and introduce again a nonzero element f in position (3, 2) and (2, 3) in $\tilde{D}$, and a nonzero element in positions (4, 2) and (2, 4) in $\tilde{T}$. By another corresponding elementary transformation E_3, we annihilate the element f in positions (3, 2) and (2, 3) in $\tilde{D}$ again and do not introduce any new nonzero elements in $\tilde{T}$. That is, as each element x_2 in positions (i, j) and (j, i) is annihilated by an orthogonal transformation $G_{i,j}$, it produces a nonzero element f in positions (i, i-1) and (i-1, i) in $\tilde{D}$, which is immediately annihilated by a suitably chosen E_i. In the same way as discussed in <u>STEP I</u>, a corresponding scaling matrix S_i may be required prior to performing $G_{i,j}$ to insure that the future multiplier p in E_i will satisfy $|p| \leq 1$ for numerical stability.

The process continues in a similar way, chasing the unwanted nonzero elements down to the bottom, right-hand corner. It ends with $M_n = E_n G_{n,n-2} S_n$ where n is the current order of matrices $\tilde{T}$ and $\tilde{D}$, and we have obtained a new tridiagonal and a diagonal matrices again. If this process is applied iteratively with properly chosen origin shifts, there result sequences of matrices $T^{(1)}$, $T^{(2)}$, ..., and $D^{(1)}$, $D^{(2)}$, ..., satisfying

$$T^{(i+1)} = Z^{(i)} T^{(i)} Z^{(i)^t}, \text{ and } D^{(i+1)} = Z^{(i)} D^{(i)} Z^{(i)^t}, \qquad (13)$$

where $$Z^{(i)} = M_n^{(i)} M_{n-1}^{(i)} \cdots M_4^{(i)} M_3^{(i)} M_2^{(i)},$$

$$M_j^{(i)} = E_j^{(i)} G_{j,\, j-2}^{(i)} S_j^{(i)}, \text{ for } j = 3, 4, \ldots, n \qquad (14)$$

and the special transformation,

$$M_2^{(i)} = E_2^{(i)} G_{2,1}^{(i)} S_2^{(i)}$$

which is produced from the origin shift. The matrix $T^{(i)}$ will approach diagonal after several iterations, replacing the negligible subdiagonal elements of $T^{(i)}$ by zeros. So the eigenvalues are given by,

$$\lambda_j = \frac{\alpha_j}{\beta_j}, \text{ if } \beta_j \neq 0$$

where $T^{(i)} = \text{diag}[\alpha_j]$ and $D^{(i)} = \text{diag}[\beta_j]$. It can be shown that one iteration requires roughly 22n multiplications (excluding scaling) and n square roots.

To replace the negligible subdiagonal elements in T by zeros, we use the following two criteria, consider any 2×2 corresponding diagonal submatrices in T and D, assume that these are as given in (8). From (10) we see that β_2 is negligible if,

1) $|\gamma| \leq \varepsilon_M(|\tau_1|+|\tau_2|)$ is satisfied, where ε_M = machine epsilon, or

2) both $\frac{|\gamma|}{|\tau_1|+|\tau_2|} \leq 16\varepsilon_M$, and

$$\left|\frac{\gamma}{\tau_1\tau_2}\right| \leq \varepsilon_M \text{ are satisfied.}$$

The eigenvalues of

$$\begin{bmatrix} \tau_1 & \gamma \\ \gamma & \tau_2 \end{bmatrix} y = \lambda y$$

are given by,

$$\lambda^2 - \lambda(\tau_1 + \tau_2) + \tau_1\tau_2(1 - \frac{\gamma^2}{\tau_1\tau_2}) = 0$$

thus, if the term $\frac{\gamma^2}{|\tau_1\tau_2|} \leq \varepsilon_M$, i.e. negligible, then our origin shift will be just the same as if $\beta_2 \equiv 0$ in T.

One special case may be disposed of immediately, since we assume that the matrix D is positive definite and its diagonal elements are in ascending order, so it may happen that D has d_{11} so small that dividing by it may cause overflow. In this case we use the following criterion: If $d_{11} < \varepsilon_M$ is satisfied, then replace d_{11} by 0. Thus T and D are of the form,

$$T = \begin{bmatrix} x_1 & x_2 & & & \\ x_2 & x & x & & \\ & \ddots & \ddots & \ddots & \\ & & x & x & x \\ & & & x & x \end{bmatrix}, \text{ and } D = \begin{bmatrix} 0 & & & \\ & d_{22} & & \\ & & \ddots & \\ & & & d_{nn} \end{bmatrix}.$$

In the above situation we use an elementary transformation E_2 to annihilate the element x_2 in positions (2, 1) and (1, 2) in T, then the matrices $T' = E_2^t T E_2$ and $D' = E_2^t D E_2$ are given by,

$$T' = \begin{bmatrix} x & 0 & & & \\ 0 & x & x & & \\ & x & \ddots & \ddots & x \\ & & \ddots & x & x \end{bmatrix} \text{ and } D' = \begin{bmatrix} 0 & & & \\ & x & & \\ & & x & \\ & & & \ddots \; x \end{bmatrix},$$

where the multiplier of E_2^t is $p = \frac{-x_2}{x_1}$. If $|p| > 1$, we just scale both T and D by the matrix $S_1 = \text{diag}(q, 1, \ldots, 1)$ with $q = 16^m$, where m is chosen as the smallest integer such that

$$\left|\frac{x_2}{x_1 q}\right| \leq 1$$

i.e., $$\left[\frac{\log_e |p|}{\log_e 16}\right] \leq m.$$

Now we go back to our original generalized eigenvalue problem $Ax = \lambda Bx$. As we have mentioned in section 2.1 above, that before we deal with the problem $\tilde{A}\tilde{x} = \lambda\tilde{D}\tilde{x}$, we have to diagonalize the matrix B first. This diagonalization can be done by applying our SQZ algorithm to the standard eigenvalue problem $B\hat{x} = \mu\hat{x}$ to solve for the eigenvalues and eigenvectors of B. Then those eigenvalues are the diagonal elements of $\tilde{D}$. Of course, every transformation which is applied to B in solving $B\hat{x} = \mu\hat{x}$ must also be applied to the matrix A, i.e. if $ZBZ^t = \tilde{D}$, then $\tilde{A} = ZAZ^t$. Therefore, the gen-

eralized eigenvalue problem $Ax = \lambda Bx$ is reduced to $\tilde{A}(Z^{-t}x) = \lambda\tilde{D}(Z^{-t}x)$, i.e. $\tilde{A}\tilde{x} = \lambda\tilde{D}\tilde{x}$, where $\tilde{x} = Z^{-t}x$. The problem $\tilde{A}\tilde{x} = \lambda\tilde{D}\tilde{x}$ can then be solved as we have discussed above.

Note that only orthogonal transformations are applied in the process of solving for the eigenvalues and eigenvectors of B.

It is interesting to compare the number of operations required to solve the eigenvalue problem $Ax = \lambda Bx$ by the SQZ and QZ algorithms where A and B are real symmetric and B is positive definite. We will assume that the QZ will be applied without modification so in reducing B to the upper triangular form the symmetry of A is destroyed. For the SQZ algorithm, assuming that we need both the eigenvalues and eigenvectors, the operation count can be divided as follows:

(i) $B\hat{x} = \mu\hat{x}$:

1. In reducing B to tridiagonal form we require roughly $\frac{4}{3}n^3$ multiplications and $\frac{1}{2}n^2$ square roots.

2. Reducing the resulting tridiagonal matrix to the diagonal form essentially is an iterative process that requires 17n multiplications and n square roots per iteration.

3. Forming the corresponding matrix $\tilde{A}$ consists of the following three parts:

(a) Saving all the orthogonal transformations in 1 above requires roughly $2n^3$ multiplications.

(b) Saving all the orthogonal transformations in 2 above requires roughly $4n^2$ multiplications per iteration.

(c) Forming $XAX^t = \tilde{A}$, where X is the orthogonal transformation formed in (a) and (b) above, requires roughly $2n^3$ multiplications.

(ii) $\tilde{A}x = \lambda\tilde{D}x$:

1. In reducing $\tilde{A}$ to the tridiagonal form T and preserving the diagonal $\tilde{D}$, we require $\frac{4}{3}n^3$ multiplications and $\frac{1}{2}n^2$ square roots.

2. Reducing the resulting tridiagonal matrix T to the diagonal form and preserving the diagonal D essentially is an iterative process again. In this reduction step we require 22n multiplications and n square roots per iteration.

3. Forming the eigenvector matrix Z consists of the following two parts:

(a) Saving all the orthogonal transformations in 1 requires roughly $2n^3$ multiplications.

(b) Saving all the orthogonal transformations in 2 requires roughly $4n^2$ multiplications per iteration.

The operation counts given above can be summarized as follows:

TABLE 2-1

OPERATION COUNT OF REDUCTION STEP
(For SQZ it consists of the steps (i), (ii)-1, and (ii)-3.a given above.)

	without eigenvectors		with eigenvectors		
	*	$\sqrt{\ }$	*	$\sqrt{\ }$	
SQZ	$\frac{20}{3}n^3 + 4n^2k$	$n^2 + nk$	$\frac{26}{3}n^3 + 4n^2k$	$n^2 + nk$	(*)
QZ	$\frac{17}{3}n^3$	n^2	$\frac{43}{6}n^3$	n^2	

(*) k is the number of iterations required in step (i)-2.

TABLE 2-2

OPERATION COUNT FOR ONE ITERATION
(In SQZ we have $Ty=\lambda Dy$ where T is tridiagonal and D diagonal, in QZ we have $Hy=\lambda Ry$ where H is an upper Hessenberg and R upper triangular.)

	without eigenvectors		with eigenvectors	
	$*$	$\sqrt{\ }$	$*$	$\sqrt{\ }$
SQZ	$22n$	n	$4n^2$	n
QZ	$13n^2$	$3n$	$21n^2$	$3n$

If $k = 2n$ and assuming that we need two iterations per eigenvalue λ in Table 3-2, then SQZ will require a total of:

Without Eigenvectors	With Eigenvectors
$\frac{88}{6} n^3$	$\frac{148}{6} n^3$

multiplications while QZ will require:

Without Eigenvectors	With Eigenvectors
$\frac{190}{6} n^3$	$\frac{295}{6} n^3$

multiplications, where we assumed that n is so large that n and n^2 are negligible compared to n^3.

CHAPTER 3

NUMERICAL RESULTS

The SQZ algorithm has been implemented in a FORTRAN program. The program is written for finding the solutions of the generalized eigenproblem $Ax = \lambda Bx$ for real symmetric matrices. Generally, we call our SQZ program twice for the full symmetric matrices A and B. The first call of SQZ reduces B to a diagonal matrix and updates the symmetric matrix A. If B is already in diagonal form, then the first call of SQZ program is not necessary. The second call of SQZ program finds the eigenvalues and, if required, the eigenvectors of the system. Between the first and the second calls, we rearrange the diagonal elements of B such that they are in ascending order to yield better results. All the examples presented in this paper have been run on the IBM 360/75 at the University of Illinois.

In all the examples we have seen so far, our SQZ algorithms require roughly 1.3 iterations per eigenvalue. Also, we tend to find the smaller eigenvalues first so we can guarantee that the smaller eigenvalues computed by SQZ will be as accurate as possible.

We compare some results obtained by <u>SQZ</u>, <u>QZ</u>, and <u>TRED2 and IMTQL2 [7]</u> applied to $B^{-1/2}AB^{-1/2}$. The program IMTQL2 or program IMTQL1 may yield better results if we arrange the diagonal elements of T, the tridiagonal matrix obtained from $B^{-1/2}AB^{-1/2}$, in ascending order. Thus for those examples which produce the diagonal elements of T in descending order, we flip the matrix T before we call IMTQL1 or IMTQL2 to guarantee better re-

sults. Also, for some other examples, we compare some results obtained by SQZ, QZ, and Kahan and Varah's program "RECURSECTION" [5].

In the SQZ program we actually combine the orthogonal transformation $G_{i,j}$ and the corresponding elementary transformation E_i as a single transformation as follows:

$$E_i G_{i,j} = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & p & \\ & & 0 & 1 & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & c & s & \\ & & s & -c & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & cl & sl & \\ & & s & -c & \\ & & & & \ddots \\ & & & & & 1 \end{bmatrix},$$

where $cl = c + ps$, and $sl = s - pc$.

Furthermore, we print out the normalized eigenvectors in our program output. And the relative residual is computed as,

$$\frac{||\beta_i Ax - \alpha_i Bx||_\infty}{(|\beta_i| \; ||A||_\infty + |\alpha_i| \; ||B||_\infty)}$$

where α_i is the i^{th} diagonal element of the final diagonal form of the matrix A and β_i is the i^{th} diagonal element of the corresponding diagonal matrix B. The i^{th} eigenvalue is given by $\frac{\alpha_i}{\beta_i}$. Numerous examples had been tested using SQZ and most of them gave results agreeing to at least 14 significant figures with QZ, TRED2 and IMTQL2, or RECURSECTION [5] (their ALGOL program has been translated into FORTRAN on the IBM 360/75 at the University of Illinois by the author). The following carefully chosen examples, however, indicate some interesting disagreement in the results obtained by SQZ, and QZ. For those underlined eigenvalues we find that there is more agreement between SQZ and TRED2 and IMTQL2 (or TRED1 and IMTQL1), while QZ gives a poorer result.

Example 1:

$$A = \begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & -3 & 1 & 5 \\ 3 & 1 & 6 & -2 \\ 4 & 5 & -2 & -1 \end{bmatrix}, \quad B = \mathrm{diag}(10^{-8}, 10^{-4}, 1, 10^{8})$$

	Result	Residual	Number of Iterations
(a) SQZ:	2.00005 00362 50999 $\times 10^{8}$	10^{-21}	0
	-3.49991 96525 38813 $\times 10^{4}$	10^{-18}	1
	$\underline{1.57142\ 57845\ 877}48$ $\times 10^{0}$	10^{-16}	0
	-5.16363 48781 31513 $\times 10^{-7}$	10^{-19}	2
(b) QZ:	∞	10^{-16}	0
	-3.50000 71425 3903 $\times 10^{4}$	10^{-17}	0
	-5.16363 48717 6210 $\times 10^{-7}$	10^{-12}	0
	$\underline{1.5}6603\ 53712\ 4899$ $\times 10^{0}$	10^{-11}	2
(c) TRED1 and IMTQL1:	-5.16363 48781 31478 $\times 10^{-7}$		2
	$\underline{1.57142\ 57845\ 877}53$ $\times 10^{0}$		1
	-3.49991 96525 38812 $\times 10^{4}$		1
	2.00005 00362 51001 $\times 10^{8}$		0

Example 2:

$$A = \begin{bmatrix} 6 & 4 & 4 & 1 \\ 4 & 6 & 1 & 4 \\ 4 & 1 & 6 & 4 \\ 1 & 4 & 4 & 6 \end{bmatrix}, \quad B = \mathrm{diag}(10^{-8}, 10^{-4}, 1, 10^{8})$$

	Result				Residual	Number of Iterations
(a) SQZ:	6.00026	67081	46831	$\times 10^{8}$	10^{-22}	0
	3.33326	85379	42870	$\times 10^{4}$	10^{-19}	0
	2.49993	75718	79956	$\times 10^{0}$	10^{-17}	1
	-7.49999	96999	70010	$\times 10^{-8}$	10^{-21}	2
(b) QZ:	∞				10^{-16}	0
	3.33341	66729	2031	$\times 10^{4}$	10^{-17}	0
	-7.49999	96139	4273	$\times 10^{-8}$	10^{-10}	0
	2.10486	86976	6099	$\times 10^{0}$	10^{-10}	2
(c) TRED1 and IMTQL1:	-7.49999	96999	69996	$\times 10^{-8}$		2
	2.49993	75718	79958	$\times 10^{0}$		1
	3.33326	85379	42865	$\times 10^{4}$		1
	6.00026	67081	46830	$\times 10^{8}$		0

Example 3:

$$A = \begin{bmatrix} 2 & 1.41421 & & \\ 1.41421 & -5.98802 & -4.31621 & \\ & -4.31621 & -2.51908 & 1.38862 \\ & & 1.38862 & 4.48198 \end{bmatrix}, \quad B = \mathrm{diag}(10^{-8},\ 1.998206\times 10^{-4},\ 9.338338\times 10^{-2},\ 1.674716\times 10^{7}).$$

	Result				Residual	Number of Iterations
(a) SQZ:	2.00005	00358	89225	$\times 10^{8}$	10^{-25}	0
	-3.49991	16417	62914	$\times 10^{4}$	10^{-18}	1
	1.57143	62474	91697	$\times 10^{0}$	10^{-15}	0
	-5.16353	76268	49124	$\times 10^{-7}$	10^{-18}	2
(b) QZ:	2.00005	00358	8923	$\times 10^{8}$	10^{-31}	0
	-3.49991	16417	6292	$\times 10^{4}$	10^{-27}	0
	-5.16353	76243	1448	$\times 10^{-7}$	10^{-12}	0
	1.56990	86436	5441	$\times 10^{0}$	10^{-12}	3

(c) IMTQL1:	-5.16353	76268	49086	$\times 10^{-7}$	2
	<u>1.57143 62474 91</u>707			$\times 10^{0}$	1
	-3.49991	16417	62914	$\times 10^{4}$	1
	2.00005	00358	89226	$\times 10^{8}$	0

Example 4:

$$A = \begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & -3 & 1 & 5 \\ 3 & 1 & 6 & -2 \\ 4 & 5 & -2 & -1 \end{bmatrix}, \quad B = P^t D P$$

where

$$P = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

and $D = \mathrm{diag}(8\times 10^8, 7, 10^{-4}, 2\times 10^{-8})$.

	Result				Residual	Number of Iterations
(a) SQZ:	2.28911	60739	92863	$\times 10^{6}$	10^{-14}	0
	-2.26278	38980	36449	$\times 10^{6}$	10^{-14}	1
	-8.57121	08967	89169	$\times 10^{-1}$	10^{-17}	1
	1.31481	48112	63813	$\times 10^{-8}$	10^{-18}	1
(b) QZ:	-1.08900	87389	8385	$\times 10^{7}$	10^{-16}	0
	∞				10^{-16}	0
	-8.57121	08547	2372	$\times 10^{-1}$	10^{-16}	0
	1.31481	48112	6382	$\times 10^{-8}$	10^{-16}	3

(c) TRED2 and IMTQL2:	1.31481	48112	63815	$\times 10^{-8}$	2
	-8.57121	07213	74638	$\times 10^{-1}$	1
	8.57469	90230	68577	$\times 10^{5}$	1
	-8.56361	98936	36599	$\times 10^{5}$	0

Example 5:

$$A = \begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & -3 & 1 & 5 \\ 3 & 1 & 6 & -2 \\ 4 & 5 & -2 & -1 \end{bmatrix}, \quad B = P^t D P,$$

where

$$P = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{bmatrix},$$

and $D = \mathrm{diag}(2\times 10^{10},\ 7\times 10^{6},\ 9\times 10^{2},\ 3\times 10^{-10})$.

	Result				Residual	Number of Iterations
(a) SQZ:	1.48710	73189	34508	$\times 10^{2}$	10^{-14}	0
	-1.48699	49075	32303	$\times 10^{2}$	10^{-14}	1
	-8.57039	53842	65270	$\times 10^{-7}$	10^{-15}	0
	5.25869	12660	45756	$\times 10^{-10}$	10^{-16}	2
(b) QZ:	-1.00251	66044	8551	$\times 10^{2}$	10^{-16}	0
	1.00258	55293	4470	$\times 10^{2}$	10^{-16}	0
	-8.57039	53838	5853	$\times 10^{-7}$	10^{-16}	0
	5.25869	12662	9592	$\times 10^{-10}$	10^{-17}	2

(c) TRED2 and IMTQL2:	$5.25869\ 12662\ 95921 \times 10^{-10}$		2
	$-8.57039\ 53838\ 65502 \times 10^{-7}$		0
	$\underline{9.44399\ 35712\ 69104 \times 10^{1}}$		1
	$\underline{-9.44396\ 95971\ 92862 \times 10^{1}}$		0

For these underlined eigenvalues, there is only one significant figure agreement among all three methods while the other eigenvalues agree as expected.

<u>Example 6</u>:

$$A = \begin{bmatrix} 10 & 1 & & & & & & & & & \\ 1 & 9 & 1 & & & & & & & & \\ & 1 & 8 & 1 & & & & & & & \\ & & \ddots & \ddots & \ddots & & & & & & \\ & & & 1 & -9 & 1 & & & & & \\ & & & & 1 & -10 & 1 & & & & \\ & & & & & 1 & -9 & 1 & & & \\ & & & & & & \ddots & \ddots & \ddots & & \\ & & & & & & & 1 & 8 & 1 & \\ & & & & & & & & 1 & 9 & 1 \\ & & & & & & & & & 1 & 10 \end{bmatrix}_{41 \times 41}$$

$B = \mathrm{diag}(1, 2, 3, \ldots, 37, \varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$

where
$$\varepsilon_1 = 16 \times 10^{-10}$$
$$\varepsilon_2 = 16 \times 10^{-11}$$
$$\varepsilon_3 = 16 \times 10^{-12}$$
$$\varepsilon_4 = 16 \times 10^{-13}$$

(a) SQZ:	$6.25685\ 99337\ 20476\times10^{12}$	$-3.45956\ 99738\ 86960\times10^{-1}$
	$5.56410\ 58478\ 29690\times10^{11}$	$-2.39999\ 08392\ 44068\times10^{-1}$
	$4.93162\ 79164\ 70630\times10^{10}$	$-2.91631\ 66491\ 44512\times10^{-1}$
	$4.28820\ 23318\ 53898\times10^{9}$	$-3.12483\ 68349\ 82157\times10^{-1}$
	$1.82853\ 14710\ 09405\times10^{-1}$	$-2.66666\ 06521\ 50588\times10^{-1}$
	$1.45237\ 26693\ 97532\times10^{-1}$	$-2.14285\ 70906\ 63091\times10^{-1}$
	$1.15479\ 41407\ 49982\times10^{-1}$	$-1.53846\ 15381\ 95596\times10^{-1}$
	$8.83407\ 59855\ 16290\times10^{-2}$	$-8.33333\ 33333\ 25440\times10^{-2}$
	$6.06101\ 16889\ 29363\times10^{-2}$	$\underline{-3.32243\ 68998\ 89234\times10^{-14}}$
	$3.12500\ 64277\ 98450\times10^{-2}$	$\underline{3.40489\ 59416\ 90256\times10^{-14}}$
	$-5.36158\ 24036\ 53544\times10^{-1}$	$1.00000\ 00000\ 00102\times10^{-1}$
	$-3.33333\ 33348\ 39045\times10^{-2}$	$2.22222\ 22222\ 24368\times10^{-1}$
	$-4.85622\ 51254\ 94727\times10^{-1}$	$3.75000\ 00000\ 45261\times10^{-1}$
	$-6.89655\ 17241\ 44940\times10^{-2}$	$5.71428\ 57152\ 66886\times10^{-1}$
	$-4.43604\ 16099\ 62082\times10^{-1}$	$8.33333\ 33554\ 70959\times10^{-1}$
	$-1.07142\ 85713\ 30581\times10^{-1}$	$1.20000\ 00527\ 48301\times10^{0}$
	$-4.14111\ 19392\ 21584\times10^{-1}$	$1.75000\ 13594\ 80318\times10^{0}$
	$-3.89489\ 14106\ 95404\times10^{-1}$	$2.66670\ 62771\ 92957\times10^{0}$
	$-1.48148\ 14764\ 94751\times10^{-1}$	$4.50143\ 82241\ 27761\times10^{0}$
	$-1.92307\ 66966\ 06680\times10^{-1}$	$1.00898\ 09703\ 94306\times10^{1}$
	$-3.54664\ 61165\ 05542\times10^{-1}$	

(b) The corresponding results in QZ and RECURSECTION agree to at least 14 significant figures with the results from SQZ shown above, except for the two smallest underlined eigenvalues as expected. The largest residual computed in SQZ is 10^{-10} while that of the QZ is 10^{-15}, the difference is due to the fact that QZ uses orthogonal transformations all the way while SQZ uses also elementary transformations which may affect the accuracy of the computed eigenvectors.

LIST OF REFERENCES

[1] R. S. Martin and J. H. Wilkinson, "Reduction of the Symmetric Eigenproblem $Ax = \lambda Bx$ and Related Problems to Standard Form," Numerische Mathematik, 11 (1968), 99-110.

[2] G. Peters and J. H. Wilkinson, "Eigenvalues of $Ax = \lambda Bx$ with Band Symmetric A and B," Comput. J., 12 (1969), 398-404.

[3] G. W. Stewart, "On the Sensitivity of the Eigenvalue Problem $Ax = \lambda Bx$," SIAM J. Numer. Anal., 9, 4 (December, 1972).

[4] G. Peters and J. H. Wilkinson, "$Ax = \lambda Bx$ and the Generalized Eigenproblem," SIAM J. Numer. Anal., 7, 4 (December, 1970).

[5] W. Kahan and J. Varah, "Two Working Algorithms for the Eigenvalues of A Symmetric Tridiagonal Matrix," Technical Report No. CS43, August 1966, Computer Science Department, School of Humanities and Sciences, Stanford University, Stanford, California.

[6] G. Fix and R. Heiberger, "An Algorithm for the Ill-conditioned Generalized Eigenvalue Problem," SIAM J. Numer. Anal., 9, 1 (March, 1972).

[7] A. Dubrulle, R. S. Martin and J. H. Wilkinson, "The Implicit QL Algorithm," Numer. Math., 12 (1968), 377-383.

[8] H. Bowdler, R. S. Martin, C. Reinsch and J. H. Wilkinson, "The QR and QL Algorithms for Symmetric Matrices," Numer. Math., 11 (1968), 293-306.

[9] C. B. Moler and G. W. Stewart, "An Algorithm for the Generalized Matrix Eigenvalue Problem," Report of University of Texas or Stanford University.

[10] J. H. Wilkinson, The Algebraic Eigenvalue Problem. Oxford: Oxford University Press, 1965.

[11] J. G. Francis, "The QR Transformation--an unitary analogue to the LR transformation," Computer Journal, 4, 265-271, 332-345, 1961-1962.

APPENDIX:

FORTRAN PROGRAM

```
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL WANTX
      DIMENSION A(41,41),B(41,41),X(41,41),SAB(820),ALFA(41),BETA(41)
      DIMENSION AD(41),BD(41),D(41),ITER(41),WI(41)
      EQUIVALENCE (A,B),(D,BETA)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C   THIS PROGRAM IS WRITTEN TO SOLVE THE GENERALIZED EIGENVALUE PROBLEM,
C   A*X=LAMBDA*B*X, WHERE A AND B ARE REAL SYMMETRIC MATRICES AND B IS AN
C   ILL-CONDITIONED POSITIVE DEFINITE MATRIX. THIS PROGRAM IS A SYMMETRIC
C   CASE OF MOLER AND STEWART'S QZ PROGRAM [9] BUT IT PRESERVES SYMMETRY,
C   REDUCES THE STORAGE REQUIREMENTS, USES LESS TIME, AND PRODUCES ONLY
C   REAL EIGENVALUES.
C
C   USAGE :
C         REAL*8 A(ND,ND),B(ND,ND),X(ND,ND),SAB(ND*(ND-1)/2),AD(ND),BD(ND)
C         REAL*8 ALFA(ND),BETA(ND),D(ND),WI(ND),ITER(ND)
C         EQUIVALENCE (A,B),(D,BETA)
C            .
C            .
C            .
C         CALL SQZ(ND,N,B,D,EPS,.TRUE.,X,&2,IERR,&4,ITER,IFLAG)
C            .
C            .
C            .
C         CALL SQZ(ND,N,A,D,EPS,WANTX,X,&3,IERR,&4,ITER,IFLAG)
C            .
C            .
C            .
C         END
C     WHERE THE PARAMETERS USED ARE :
C     INPUT :
C        A, B       - CONTAIN THE N BY N REAL SYMMETRIC MATRICES AND USE
C                     THE SAME ARRAY STORAGE INDEPENDENTLY IN 1-ST AND
C                     2-ND CALLING OF SUBROUTINE SQZ.
C        SAB        - IS A VECTOR WHICH HAS ND*(ND-1)/2 ELEMENTS AND
C                     IS USED TO SAVE THE ORIGINAL INFORMATION OF ONE OF
C                     THE TWO SYMMETRIC MATRICES A AND B WHILE THE OTHER
C                     IS STORED IN THEIR SHAREING ARRAY STORAGE.
C        N          - ORDER OF THE INPUT MATRICES.
C        ND         - NUMBER OF ROWS OR COLUMNS OF ARRAYS SPECIFIED.
C        D          - CONTAINS THE DIAGONAL ELEMENTS OF IDENTITY MATRIX FOR
C                     1-ST CALL OF SQZ, AND THE EIGENVALUES OF MATRIX B FOR
C                     2-ND CALL OF SQZ. THIS VECTOR USES THE SAME ARRAY
C                     STORAGE OF VECTOR BETA(ND).
C        AD         - CONTAINS THE DIAGONAL ELEMENTS OF ORIGINAL DATA OF A.
C        BD         - CONTAINS THE DIAGONAL ELEMENTS OF ORIGINAL DATA OF B.
C        X          - CONTAINS N BY N IDENTITY MATRIX FOR 1-ST CALL OF SQZ,
C                     AND THE TRANSFORMATION MATRICES IN DIAGONALIZING  B
C                     FOR 2-ND CALL OF SQZ.
C        EPS        - IS THE RELATIVE PRECISION OF ELEMENTS OF  A AND B.
C        IFLAG      - INDICATES THE REQUIREMENT OF PERFORMING ELEMENTARY
C                     TRANSFORMATIONS,
C                        IFLAG=0 - INDICATES ELEMENTARY TRANSFORMATIONS ARE
C                                  REQUIRED,
C                        IFLAG=1 - INDICATES NO ELEMENTARY TRANSFORMATIONS
C                                  ARE REQUIRED.
C        WANTX      - INDICATES THE REQUIREMENT OF COMPUTING EIGENVECTORS,
C                        WANTX=.TRUE. - MEANS EIGENVECTORS ARE REQUIRED,
C                        WANTX=.FALSE.- MEANS EIGENVECTORS ARE NOT REQUIRED.
C     OUTPUT :
C        ALFA,BETA  - ARE VECTORS AND CONTAIN THE INFORMATION OF EIGEN-
```

```
C                        VALUES. THE I-TH EIGENVALUE IS GIVEN BY,
C                          LAMBDA(I) = ALFA(I)/BETA(I).
C        A, B          - THE STRICTLY UPPER TRIANGLE(I.E. A(I,J)=0. FOR I>=J)
C                        CONTAINS THE ORIGINAL INFORMATION AND IS SAVED THROU-
C                        GHOUT THE COMPUTATION BUT THE LOWER TRIANGLE IS
C                        OVERWRITTEN AND ONLY THE DIAGONAL WHICH CONTAINS THE
C                        VALUES OF ALFA(I) IS NEEDED ON OUTPUT, THE STRICTLY
C                        LOWER TRIANGLE IS DESTROYED.
C        D             - CONTAINS THE VALUES OF BETA(I).
C        X             - CONTAINS THE EIGENVECTORS IF REQUIRED.
C        ITER          - ITER(I) CONTAINS THE NUMBER OF ITERATIONS NEEDED FOR
C                        FINDING THE I-TH EIGENVALUE.
C        IERR          - IS TROUBLE INDICATOR,
C                          IERR=J - INDICATES THE J-TH EIGENVALUE HAS NOT BEEN
C                                   DETERMINED AFTER CERTAIN AMOUNT OF NUMBER
C                                   OF ITERATIONS(WE SET J=30 IN THIS PROGRAM)
C        &J            - INDICATES SPECIAL RETURN POINT, J, FOR COMMENTS ON
C                        TROUBLES.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C   IBM STANDARD FIXUP SUBROUTINE OF ERROR #207 & #208 :
        CALL ERRSET(207,256,-1,1,0,208)
      WANTX=.TRUE.
      EPS=2.22D-16
      ND=41
    1 CONTINUE
      READ(5,10,END=996) N
      WRITE(6,9) N
    9 FORMAT(1H1,' N =',I3//,' B =')
   10 FORMAT(I2)
      DO 6 I=1,N
      READ(5,900)    (B(I,J),J=1,N)
    6 WRITE(6,1000) (B(I,J),J=1,N)
  900 FORMAT(8D10.2)
      ANORM=0.
      BNORM=0.
      IJ=0
      DO 11 I=1,N
      D(I)=1.D0
      BNI=0.
      DO 20 J=1,N
      BNI=BNI+DABS(B(I,J))
C ---INITIALIZE X :
   20 X(I,J)=0.
      BNORM=DMAX1(BNORM,BNI)
      BD(I)=B(I,I)
      X(I,I)=1.D0
      IP1=I+1
      IF(I.GE.N) GO TO 11
      DO 21 J=IP1,N
      IJ=IJ+1
   21 SAP(IJ)=B(I,J)
   11  CONTINUE
 1000 FORMAT(' ',1P12D11.3)
      NM1=N-1
      N2=N/2
      IFLAG=1
      CALL SQZ(ND,N,B,D,EPS,.TRUE.,X,&2,IERR,&4,ITER,IFLAG)
      IFLAG=0
      PRINT 800
```

```
  800 FORMAT(/,' A =')
         DO 24 I=1,N
   24    D(I)=B(I,I)
         DO 25 I=1,N
         READ(5,900)  (A(I,J),J=1,N)
         WRITE(6,1000)(A(I,J),J=1,N)
   25    CONTINUE
      DO 27 I=1,N
      ANI=0.
      AD(I)=A(I,I)
      DO 26 J=1,N
   26 ANI=ANI+DABS(A(I,J))
   27 ANORM=DMAX1(ANORM,ANI)
      CALL ORDER(N,ND,D,X)
      PRINT 700, (D(I),I=1,N)
  700 FORMAT(/,' EIGENVALUES OF MATRIX  B ='/,(4X,1P5D25.16))
C ---PERFORM TRANSFORMATIONS ON MATRIX A :
      CALL XTCX(ND,N,A,X,AD,W1)
      CALL SQZ(ND,N,A,D,EPS,WANTX,X,&3,IERR,&4,ITER,IFLAG)
      DO 35 I=1,N
   35 ALFA(I)=A(I,I)
      NM1=N-1
        IF(.NOT.WANTX) GO TO 32
C --- RELOAD  B  &  SAVE  A :
      IJ=0
      DO 40 I=1,N
      IP1=I+1
      IF(I.GE.N) GO TO 41
      DO 40 J=IP1,N
      IJ=IJ+1
      T=A(I,J)
      B(I,J)=SAB(IJ)
   40 SAB(IJ)=T
   41 CONTINUE
      CALL XTCX(ND,N,B,X,BD,W1)
   32 CONTINUE
      DO 997 I=1,N
      IF (BETA(I).EQ.0.) GO TO 999
      EIG=ALFA(I)/BETA(I)
      WRITE(6,998) EIG,ALFA(I),BETA(I)
  998 FORMAT(/' LAMBDA= ',1PD26.16,/' ','ALFA,BETA=',1P2D26.16,
     +     /3X,'EIGENVECTOR=')
      GO TO 150
  999 WRITE(6,995) ALFA(I),BETA(I)
  995 FORMAT(/' ','LAMBDA= ** INFINITE ***'/,' ','ALFA,BETA=',1P2D26.16,
     +     /3X,'EIGENVECTOR=')
  150 CONTINUE
      IF(.NOT.WANTX) GO TO 997
      W1(I)=DSQRT(DABS(B(I,I)))
      DO 33 K=1,N
   33 X(K,I)=X(K,I)/W1(I)
      PRINT  994, (X(J,I),J=1,N)
  994 FORMAT('*',4X,1PD25.16)
         RN = 0.
      KJ=0
         DO 55 K=1,N
      KM1=K-1
      KP1=K+1
      SA=AD(K)*X(K,I)
      SB= BD(K)*X(K,I)
      IF(K.EQ.N) GO TO 51
```

```
      DO 50 J=KP1,N
      KJ=KJ+1
      SA=SA+SAB(KJ)*X(J,I)
      SB=SB+B(K,J)*X(J,I)
 50          CONTINUE
 51   CONTINUE
      KC=KJ-2.D0*(N-K)
      IF(K.EQ.1) GO TO 53
      DO 52 JJ=1,KM1
      J=KM1-JJ+1
      SA=SA+SAB(KC)*X(J,I)
      KC=KC-(N-K+JJ)
 52   SB=SB+B(J,K)*X(J,I)
 53   CONTINUE
             R = DABS(BETA(I)*SA-ALFA(I)*SB)
             RN = DMAX1(R,RN)
 55       CONTINUE
          IF (RN.NE.0.)
     1    RN = RN/(DABS(BETA(I))*ANORM+DABS(ALFA(I))*BNORM)
          PRINT 208,RN
 997  CONTINUE
 208  FORMAT( 3X,'RESIDUAL=',1PD26.16///)
      GO TO 1
    2 PRINT 221, IERR
 221  FORMAT(////10X,'**THE ',I3,'-TH EIGENVALUE OF B  HAS NOT BEEN',
     +     ' DETERMINDED AFTER 30 ITERATIONS. *****'/)
      GO TO 1
    3 PRINT 222, IERR
 222  FORMAT(////10X,'***** THE ',I3,' -TH  EIGENVALUE HAS NOT BEEN',
     +     ' DETERMINDED AFTER 30 ITERATIONS. *****'/)
      GO TO 1
    4 PRINT 223
 223  FORMAT(' *** MATRIX B IS NOT POSITIVE DEFINITE DUE TO',
     +     '  --IMPROPER INPUT DATA, OR  --ROUND OFF ERROR. ***')
      GO TO 1
 996  STOP
      END
```

```
      SUBROUTINE SQZ(ND,N,AB,D,EPS,WANTX,X,*,IERR,*,ITER,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION AB(ND,ND),D(ND),X(ND,ND),ITER(ND)
      LOGICAL WANTX
      CALL SQZTRI(ND,N,AB,D,EPS,WANTX,X,IFLAG)
      CALL SQZIT(ND,N,AB,D,EPS,EPSP,ITER,&2,IERR,WANTX,X,&3,IFLAG)
      RETURN
    2 RETURN1
    3 RETURN2
      END
```

```
C
      SUBROUTINE SQZTRI(ND,N,A,D,EPS8,WANTX,X,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL WANTX
      DIMENSION A(ND,ND),D(ND),X(ND,ND)
C
C     REDUCE A TO TRIDIAGONAL, KEEP D DIAGONAL.
      IF(N.LE.2) GO TO 170
      NM2=N-2
      DO 160 K=1,NM2
        KP1=K+1
        NK1=N-K-1
        DO 150 LB=1,NK1
          L=N-LB
          L1=L+1
      IF(A(L1,K).EQ.0.) GO TO 150
      IF(IFLAG.EQ.1) GO TO 683
C :::::: SCALING :
      IF(A(L,K).EQ.0.D0) GO TO 683
      RSCALE=D(L)/D(L1)
      TSCALE=DABS(A(L1,K)/A(L,K))
      IF(TSCALE-1.D0) 681,683,682
  681 IF(RSCALE.GT.(1.D0+TSCALE)/(TSCALE-TSCALE*TSCALE))
     +         CALL SCALE(A,D,L,L1,N , K ,ND,TSCALE,N,WANTX,X)
      GO TO 683
  682 IF(RSCALE.LT.(TSCALE-1.D0)/(1.D0+TSCALE*TSCALE))
     +         CALL SCALE(A,D,L,L1,N , K ,ND,TSCALE,N,WANTX,X)
  683 CONTINUE
      R=DSQRT(A(L,K)*A(L,K)+A(L1,K)*A(L1,K))
      R=DSIGN(R,A(L,K))
      C=A(L,K)/R
      S=A(L1,K)/R
      C1=C
      S1=S
      IF(IFLAG.EQ.1) GO TO 15
      EDN=C*S*(D(L)-D(L1))
      T=D(L1)
      D(L1)=C*C*T+S*S*D(L)
      D(L)=D(L)*T/D(L1)
      IF(DABS(EDN).LE.EPS8) GO TO 15
      P=-EDN/D(L1)
      C1=C+S*P
      S1=S-C*P
   15 CONTINUE
      DO 103 J=K,L
      T=A(L1,J)
      A(L1,J)=S*A(L,J)-C*T
  103 A(L,J)=C1*A(L,J)+S1*T
      U=C1*T+S1*A(L1,L1)
      A(L1,L1)=S*T-C*A(L1,L1)
      A(L,L)=C1*A(L,L)+S1*U
      DO 104 J=L1,N
      T=A(J,L)
      A(J,L)=C1*T+S1*A(J,L1)
  104 A(J,L1)=S*T-C*A(J,L1)
      IF(.NOT.WANTX) GO TO 105
      DO 10 J=1,N
      T=X(J,L)
      X(J,L)=C1*T+S1*X(J,L1)
   10 X(J,L1)=S*T-C*X(J,L1)
  105 CONTINUE
```

```
  150   CONTINUE                                                          SQ70292
  160 CONTINUE                                                            SQ70293
  170 CONTINUE                                                            SQ70294
      RETURN                                                              SQ70295
      END                                                                 SQ70296
```

```
C                                                                        SQZ0297
      SUBROUTINE SQZIT(ND,N,A,D,EPS,EPSB,ITER,*,IERR,WANTX,X,*,IFLAG)    SQZ0298
      IMPLICIT REAL*8 (A-H,O-Z)                                          SQZ0299
      LOGICAL WANTX                                                      SQZ0300
      DIMENSION A(ND,ND),D(ND),X(ND,ND)                                  SQZ0301
      DIMENSION ITER(ND)                                                 SQZ0302
C                                                                        SQZ0303
C  INITIALIZE ITER, COMPUTE EPSB                                         SQZ0304
      LS=1                                                               SQZ0305
C                                                                        SQZ0306
      BNORM = 0.                                                         SQZ0307
      DO 185 I=1,N                                                       SQZ0308
         ITER(I) = 0                                                     SQZ0309
      BNI=DABS(D(I))                                                     SQZ0310
         IF (BNI.GT.BNORM) BNORM = BNI                                   SQZ0311
  185 CONTINUE                                                           SQZ0312
      IF (BNORM.EQ.0.) BNORM = EPS                                       SQZ0313
      EPSB=EPS                                                           SQZ0314
C                                                                        SQZ0315
C     REDUCE A TO DIAGONAL, KEEP B DIAGONAL:                             SQZ0316
      M = N                                                              SQZ0317
  200 IF (M.LE.LS)GO TO 390                                              SQZ0318
C                                                                        SQZ0319
C     CHECK FOR CONVERGENCE OR REDUCIBILITY                              SQZ0320
C                                                                        SQZ0321
      DO 220 LP=1,M                                                      SQZ0322
         L = M+1-LP                                                      SQZ0323
      LM1=L-1                                                            SQZ0324
         IF (L.EQ.LS)GO TO 260                                           SQZ0325
C                                                                        SQZ0326
      IF(DABS(A(L,LM1)).LE.EPS) GO TO 230                                SQZ0327
      TAU1=A(LM1,LM1)                                                    SQZ0328
      TAU2=A(L,L)                                                        SQZ0329
      GAMMA=A(L,LM1)                                                     SQZ0330
      IF(IFLAG.EQ.1) GO TO 219                                           SQZ0331
      DLDL1=D(L)*D(LM1)                                                  SQZ0332
      IF(DLDL1.LT.0.) RETURN2                                            SQZ0333
      DD=DSQRT(DLDL1)                                                    SQZ0334
      IF(DLDL1.GT.0.) GO TO 214                                          SQZ0335
      GO TO 220                                                          SQZ0336
  214 CONTINUE                                                           SQZ0337
      LP1=L+1                                                            SQZ0338
      DESP1=16.D-12                                                      SQZ0339
      IF(DD.GT.DESP1) GO TO 218                                          SQZ0340
      DESP2=16.D+03                                                      SQZ0341
      DESP3=DESP2*DESP2                                                  SQZ0342
      DO 217 I=1,2                                                       SQZ0343
      LP1MI=LP1-I                                                        SQZ0344
      D(LP1MI)=D(LP1MI)*DESP3                                            SQZ0345
      IF(LP1MI.GT.LS) A(LP1MI,LP1MI-1)=A(LP1MI,LP1MI-1)*DESP2            SQZ0346
      A(LP1MI,LP1MI)=A(LP1MI,LP1MI)*DESP3                                SQZ0347
      IF(.NOT.WANTX) GO TO 217                                           SQZ0348
      DO 216 J=1,N                                                       SQZ0349
  216 X(J,LP1MI)=X(J,LP1MI)*DESP2                                        SQZ0350
  217 CONTINUE                                                           SQZ0351
      A(L,LM1)=A(L,LM1)*DESP2                                            SQZ0352
      IF(LP1.LE.N) A(LP1,L)=A(LP1,L)*DESP2                               SQZ0353
  218 CONTINUE                                                           SQZ0354
C                                                                        SQZ0355
C     FIRST STOPPING CRITERION :                                         SQZ0356
      TAU1=A(LM1,LM1)/D(LM1)                                             SQZ0357
```

```
      TAU2=A(L,L)/D(L)
      GAMMA=A(L,LM1)/DD
  219 CONTINUE
      GT=DABS(TAU1)+DABS(TAU2)
      EPSAA=EPS *GT
      IF(DABS(GAMMA).LE.EPSAA) GO TO 230
C
C     SECOND STOPPING CRITERION :
      GG=DABS(TAU1*TAU2)
      EPSI=EPS**4
      IF(GG.LE.EPSI) GO TO 220
      TAU12=GAMMA**2/DABS(TAU1*TAU2)
      IF(TAU12.LE.EPS ) GO TO 777
      GO TO 220
  777 GS=GAMMA/GT
      EPSM =EPS *(16.D0)
      IF(GS.LE.EPSM ) GO TO 230
  220 CONTINUE
  230 A(L,L-1) = 0.
      IF (L.LT.M  ) GO TO 260
      M = L-1
      GO TO 200
C
C     CHECK FOR SMALL TOP OF B
C
  260 IF (DABS( D(L) ).GT.EPSB) GO TO 300
      D(L)=0.0D0
      LP1=L+1
      LP2=L+2
      IF(LP1.GT.M) LP1=M
      IF(LP2.GT.M) LP2=M
      L1=LP1
      IF(A(LP1,L).EQ.0.0D0) GO TO 280
      P=-A(LP1,L)/A(L,L)
      IF(DABS(P).LE.1.D0) GO TO 270
        IM=IDINT(DLOG(DABS(P))/DLOG(16.D0))+1
        Q=16.D0**IM
        A(L,L)=Q*Q*A(L,L)
         A(LP1,L)=Q*A(LP1,L)
      IF(.NOT.WANTX) GO TO 269
      DO 265 I=1,N
  265 X(I,L)=Q*X(I,L)
  269 CONTINUE
      P=-A(LP1,L)/A(L,L)
  270   CONTINUE
      A(LP1,LP1)=A(LP1,LP1)+P*A(LP1,L)
      IF(.NOT.WANTX) GO TO 279
      DO 275 I=1,N
  275 X(I,LP1)=X(I,LP1)+P*X(I,L)
  279 CONTINUE
  280 L=LP1
      LS=LS+1
      GO TO 230
C
C     BEGIN ONE SQZ STEP, ITERATION STRATEGY
C
  300 M1=M-1
      LP1=L+1
      LP2=L+2
      IF(LP1.GT.M) LP1=M
      IF(LP2.GT.M) LP2=M
```

```
      LI=LP1
      ITER(M)=ITER(M)+1
      IF(ITER(M).EQ.1) GO TO 305
      IF(DABS(A(M, M1)) .LT. 0.75D0*OLD1) GO TO 305
      IF(ITER(M) .LE. 30)  GO TO 305
      IERR=M
      RETURN1
  305 CONTINUE
C
C
      IF(A(M,M1) .EQ. 0.D0) GO TO 200
C :::::: FIND THE ORIGIN SHIFT :
      DM1INV=1.D0/D(M1)
      DMINV=1.D0/D(M)
      G1=A(M1,M1)*DM1INV
      G2=A(M,M)*DMINV
      EL=DSQRT(DABS(DM1INV*DMINV)) *DABS(A(M,M1))
      P=(G2-G1)/(2.D0*EL)
        R=DSQRT(P*P+1.D0)
        IF(P .LT. 0.D0) R=-R
      SHIFT=G2+EL/(P+R)
C :::::: END OF FINDING SHIFT.
C
      IF(IFLAG.EQ.1) GO TO 583
      X1=       A(L,L) - SHIFT*D(L)
      X2=                A(LP1,L)
      IF(X2.EQ.0.D0) GO TO 280
C
C :::::: SCALING :
      IF(X1.EQ.0.D0) GO TO 583
      RSCALE=D(L)/D(LP1)
      TSCALE=DABS(X2/X1)
      IF(TSCALE-1.D0) 581,583,582
  581 IF(RSCALE.GT.(1.D0+TSCALE)/(TSCALE-TSCALE*TSCALE))
     +         CALL SCALE(A,D,L,LP1,LP2,L,ND,TSCALE,N,WANTX,X)
      GO TO 583
  582 IF(RSCALE.LT.(TSCALE-1.D0)/(1.D0+TSCALE*TSCALE))
     +         CALL SCALE(A,D,L,LP1,LP2,L,ND,TSCALE,N,WANTX,X)
  583 CONTINUE
      OLD1=DABS(A(M,M-1))
      X1=       A(L,L) - SHIFT*D(L)
      X2=A(LP1,L)
C
C     PERFORM THE SPECIAL TRANSFORMATION PRODUCED FROM THE ORIGIN SHIFT:
      CALL TRANSF(A,D,L,L,LP1,LP2,WANTX,X,X1,X2,N,M1,M,ND,EPSB,IFLAG)
      IF(LP1.GE.M) GO TO 330
      DO 360 K=LP1,M1
      KM1=K-1
      K1=K+1
      K2=K+2
      IF(KM1.LT.L) KM1=L
      IF(K2.GT.M) K2=M
      IF(A(K1,KM1) .EQ. 0.D0) GO TO 360
      IF(IFLAG.EQ.1) GO TO 683
C :::::: SCALING :
      IF(A(K,KM1).EQ.0.D0) GO TO 683
      RSCALE=D(K)/D(K1)
      TSCALE=DABS(A(K1,KM1)/A(K,KM1))
      IF(TSCALE-1.D0) 681,683,682
  681 IF(RSCALE.GT.(1.D0+TSCALE)/(TSCALE-TSCALE*TSCALE))
     +         CALL SCALE(A,D,K,K1,K2,KM1,ND,TSCALE,N,WANTX,X)
```

```
      GO TO 683                                                         SQZ0480
  682 IF(RSCALE.LT.(TSCALE-1.D0)/(1.D0+TSCALE*TSCALE))                  SQZ0481
     +        CALL SCALE(A,D,K,K1,K2,KM1,ND,TSCALE,N,WANTX,X)           SQZ0482
  683 CONTINUE                                                          SQZ0483
C                                                                       SQZ0484
C     PERFORM TRANSFORMATIONS TO ZERO A(K+1,K-1) & A(K-1,K+1) AND       SQZ0485
C     KEEP B DIAGONAL :                                                 SQZ0486
      X1=A(K,KM1)                                                       SQZ0487
      X2=A(K1,KM1)                                                      SQZ0488
      CALL TRANSF(A,D,KM1,K,K1,K2,WANTX,X,X1,X2,N,M1,M,ND,EPS3,IFLAG)   SQZ0489
  360 CONTINUE                                                          SQZ0490
  330 CONTINUE                                                          SQZ0491
C     END MAIN LOOP                                                     SQZ0492
C                                                                       SQZ0493
      GO TO 200                                                         SQZ0494
C                                                                       SQZ0495
C     END ONE SQZ STEP                                                  SQZ0496
C                                                                       SQZ0497
  390 CONTINUE                                                          SQZ0498
      WRITE(6,555)    (ITER(II),II=1,N)                                 SQZ0499
  555 FORMAT(' ****** ITER(I)=',/,(20I5))                               SQZ0500
      RETURN                                                            SQZ0501
      END                                                               SQZ0502
```

```
C
      SUBROUTINE  TRANSF(A,D,KM1,K,K1,K2,WANTX,X,X1,X2,N,M1,M,ND,EPSB,
     +                    IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL WANTX
      DIMENSION A(ND,ND),D(ND),X(ND,ND)
C
C     FIND AND PERFORM TRANSFORMATIONS TO ZERO THE UNWANTED NONZERO
C     ELEMENTS IN MATRICES A AND B :
C
      R=DSQRT(X1*X1+X2*X2)
      R=DSIGN(R,X1)
      C=X1/R
      S=X2/R
      CS=C*S
      CC=C*C
      SS=S*S
      C1=C
      S1=S
      IF(IFLAG.EQ.1) GO TO 7
      EDN=CS*(D(K)-D(K1))
        T=D(K1)
        D(K1)=CC*T+SS*D(K)
        D(K)=D(K)*T/D(K1)
      IF(DABS(EDN).LE.EPSB) GO TO 7
      P=-EDN/D(K1)
      C1=C+S*P
      S1=S-C*P
    7 CONTINUE
      DO 21 J=KM1,K
      T=A(K1,J)
      A(K1,J)=S*A(K,J)-C*T
   21 A(K,J)=C1*A(K,J)+S1*T
      IF(K.GT.KM1) A(K1,KM1)=0.
      U=C1*T+S1*A(K1,K1)
      A(K1,K1)=S*T-C*A(K1,K1)
      A(K,K)=C1*A(K,K)+S1*U
      T=A(K1,K1)
      A(K1,K1)=S*A(K1,K)-C*T
      A(K1,K)=C1*A(K1,K)+S1*T
      IF(K2.EQ.K1) GO TO 23
      A(K2,K)=S1*A(K2,K1)
      A(K2,K1)=-C*A(K2,K1)
   23 CONTINUE
      IF(.NOT.WANTX) GO TO 30
      DO 25 J=1,N
      T=X(J,K1)
      X(J,K1)=S*X(J,K)-C*T
      X(J,K)=C1*X(J,K)+S1*T
   25 CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE SCALE(A,D,K,K1,K2,KM1,ND,TSCALE,N,WANTX,X)
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL WANTX
      DIMENSION A(ND,ND),D(ND),X(ND,ND)
C
C     THIS SUBROUTINE FINDS THE SCALING FACTOR FOR STABILIZING THE
C     ELEMENTARY TRANSFORMATION :
C
      THIGH=15.D0*16.D0
      TLOW=1.D0/THIGH
      IF(TSCALE.LT.TLOW) GO TO 888
      IF(TSCALE.LE.THIGH) GO TO 889
  888 SCALE1=DSQRT(D(K1))
      SCALE2=DSQRT(D(K))
C :::::: NOW UPDATE THE ROWS AND COLUMNS K,K1 OF A AND D :
      D(K)=D(K)*D(K1)
      D(K1)=D(K)
  998 CONTINUE
      DO 12 J=KM1,K
      A(K,J)=SCALE1*A(K,J)
   12 A(K1,J)=SCALE2*A(K1,J)
      A(K,K)=SCALE1*A(K,K)
      A(K1,K1)=SCALE2*A(K1,K1)*SCALE2
      A(K1,K)=SCALE1*A(K1,K)
      IF(K2.EQ.K1) GO TO 13
      A(K2,K1)=SCALE2*A(K2,K1)
   13 CONTINUE
      IF(.NOT.WANTX) GO TO 20
      DO 14 J=1,N
      X(J,K)=SCALE1*X(J,K)
   14 X(J,K1)=SCALE2*X(J,K1)
   20 CONTINUE
      GO TO 1818
  889 SCALE1=1.D0
      SCALE2=1.D0/TSCALE
      D(K1)=(SCALE2*SCALE2)*D(K1)
      GO TO 998
 1818 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE ORDER(N,ND,D,X)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION X(ND,ND),D(ND)
C
C     THIS SUBROUTINE ORDERS THE EIGENVALUES OF MATRIX B IN ASCENDING
C     ORDER SO THAT THE BETTER RESULTS IN THE SECOND CALL OF SQZ WILL
C     BE GUARANTEED.
C
      DO 20 K=1,N
      DMIN=1.0D+38
      JMIN=K
      DO 10 J=K,N
      IF(DMIN.LE.D(J)) GO TO 10
      DMIN=D(J)
      JMIN=J
   10 CONTINUE
      T=D(JMIN)
      D(JMIN )=D(K)
      D(K)=T
      DO 20 I=1,N
      T=X(I,JMIN)
      X(I,JMIN)=X(I,K)
   20 X(I,K)=T
      RETURN
      END
```

```
      SUBROUTINE XTCX(ND,N,C,X,D,W1)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION C(ND,ND),X(ND,ND),W1(ND),D(ND)
C
C     THIS SUBROUTINE UPDATES THE MATRIX A AFTER FIRST CALL OF SOZ.
C     THIS SUBROUTINE IS ALSO USED TO PERFORM ANY MATRIX MULTIPLICATION
C     OF THE FORM   XT * Z * X.
C
C --- C * X :
      DO 90 J=1,N
      JM1=J-1
      IF(J.LE.1) GO TO 4
      DO 3 K=1,JM1
      KM1=K-1
      KP1=K+1
      W1(K)=D(K)*X(K,J)
      DO 1 I=KP1,N
    1 W1(K)=W1(K)+C(K,I)*X(I,J)
      IF(K.LE.1) GO TO 3
      DO 2 I=1,KM1
    2 W1(K)=W1(K)+C(I,K)*X(I,J)
    3 CONTINUE
    4 CONTINUE
      DO 30 K=J,N
      KM1=K-1
      KP1=K+1
      W1(K)=D(K)*X(K,J)
      IF(K.EQ.N) GO TO 15
      DO 10 I=KP1,N
   10 W1(K)=W1(K)+C(K,I)*X(I,J)
   15 CONTINUE
      IF(K.EQ.1) GO TO 30
      DO 20 I=1,KM1
   20 W1(K)=W1(K)+C(I,K)*X(I,J)
   30 CONTINUE
C --- XT * (C * X) :
      DO 50 I=J,N
      C(I,J)=0.
      DO 50 K=1,N
   50 C(I,J)=C(I,J)+X(K,I)*W1(K)
   90 CONTINUE
      RETURN
      END
```

665

```
      SUBROUTINE XTCX(ND,N,C,X,D,W1)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION C(ND,ND),X(ND,ND),W1(ND),D(ND)
C
C     THIS SUBROUTINE UPDATES THE MATRIX A AFTER FIRST CALL OF SQZ.
C     THIS SUBROUTINE IS ALSO USED TO PERFORM ANY MATRIX MULTIPLICATION
C     OF THE FORM  XT * Z * X.
C
C --- C * X :
      DO 90 J=1,N
      JM1=J-1
      IF(J.LE.1) GO TO 4
      DO 3 K=1,JM1
      KM1=K-1
      KP1=K+1
      W1(K)=D(K)*X(K,J)
      DO 1 I=KP1,N
    1 W1(K)=W1(K)+C(K,I)*X(I,J)
      IF(K.LE.1) GO TO 3
      DO 2 I=1,KM1
    2 W1(K)=W1(K)+C(I,K)*X(I,J)
    3 CONTINUE
    4 CONTINUE
      DO 30 K=J,N
      KM1=K-1
      KP1=K+1
      W1(K)=D(K)*X(K,J)
      IF(K.EQ.N) GO TO 15
      DO 10 I=KP1,N
   10 W1(K)=W1(K)+C(K,I)*X(I,J)
   15 CONTINUE
      IF(K.EQ.1) GO TO 30
      DO 20 I=1,KM1
   20 W1(K)=W1(K)+C(I,K)*X(I,J)
   30 CONTINUE
C --- XT * (C * X) :
      DO 50 I=J,N
      C(I,J)=0.
      DO 50 K=1,N
   50 C(I,J)=C(I,J)+X(K,I)*W1(K)
   90 CONTINUE
      RETURN
      END
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CAC Document No. 110	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AN ALGORITHM FOR THE SYMMETRIC GENERALIZED EIGENVALUE PROBLEM $Ax = \lambda Bx$		5. TYPE OF REPORT & PERIOD COVERED Research Report
		6. PERFORMING ORG. REPORT NUMBER CAC 110
7. AUTHOR(s) Chang-Chung Chang		8. CONTRACT OR GRANT NUMBER(s) DAHC04-72-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS Center for Advanced Computation University of Illinois at Urbana-Champaign Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA Order No. 1899
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia		12. REPORT DATE February 1974
		13. NUMBER OF PAGES 52
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) U. S. Army Research Office Box CM, Duke Station Durham, North Carolina 27706		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

No restrictions. Copies may be requested from the address in (9) above.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Approved for public release; distribution unlimited.

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Eigenvalue problem
SQZ algorithm

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An SQZ algorithm is developed, in a way similar to that of Moler and Stewart's QZ method, for handling symmetric A and B where B is an ill-conditioned positive definite matrix. This algorithm preserves symmetry, reduces the storage requirements, uses less time, and produces only real eigenvalues.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)